台科大 111 學年度「彩色影像處理」

作業三：簡化的 CNN 數值影像辨識　　孫沛立老師

附件資料:

train1000 是 1000 個 0~9 的數值影像
1~100 號：數值'0'
101~200 號：數值'1'
....
901~1000 號：數值'9'

作業內容：利用簡化的卷基神經網路(圖 1)，辨識 0 到 9 中任意兩個數值的影像資料(例如'0'與'1')。

1. 輸入要用於辨識的兩個數值(例如'0'與'1')。
2. 依序讀取附件 train1000 裡，這兩個數值所屬的影像各 100 幅。
3. 將這 200 幅影像縮小至 8x8 大小，視為訓練集，並依序執行步驟 4 至 7，產生兩層卷積與兩層池化後的特徵圖。

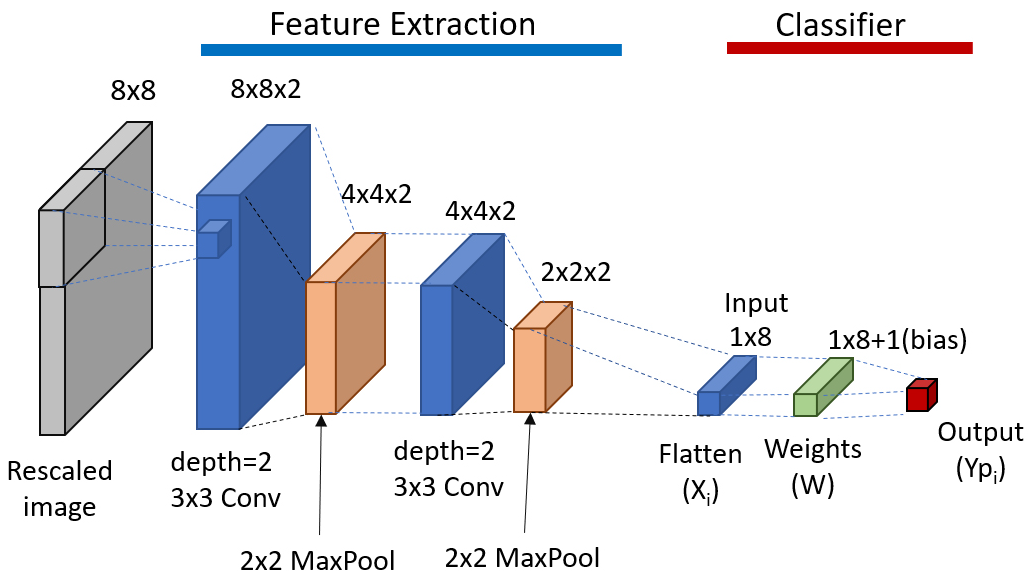

圖 1: 簡化的卷積神經網路(CNN)。

4. 第一層卷積：用兩種 3x3 濾鏡(卷積核)做卷積運算(depth=2)，運算後的特徵圖大小為 8x8x2。

$$filter1 = \begin{bmatrix} -1 & -1 & 1 \\ -1 & 0 & 1 \\ -1 & 1 & 1 \end{bmatrix}$$　　*filter*2 自行決定(可直接將 filter2 轉置)

5. 第一層池化：對步驟 2 的結果 做 2x2 的最大池化(max pooling)，運算後的特徵圖大小為 4x4x2。

   如果使用 Matlab，可先自己寫一個將 2x2 矩陣的最大值算出的 max2()函式，再用 blkproc()函式，執行分區的@max2 運算。

6. 第二層卷積：用步驟 2 的兩種 3x3 濾鏡分別對步驟 3 獲得的特徵圖的 1,2 層做卷積運算，運算後的特徵圖大小為 4x4x2。注意：CNN 實際的卷積方式不是這樣，這個作業比較接近" how_CNNs_work"講義的簡易模式。

7. 第二層池化：對步驟 6 的結果做 2x2 的最大池化(max pooling)，運算後的特徵圖大小為 2x2x2。

8. 平坦化：將 2x2x2 影像特徵轉成 1x8 的特徵資料，再加常數項 1，成為 1x9 的資料。相當於 $y=a_1x_1+ a_2x_2+ ..... a_8x_8+ a_9$，其中 y 為目標值(0 或 1 的分類標籤)，x 是影像的卷積特徵值，a 是待優化的係數。以下將透過線性迴歸，獲得係數矩陣 $A=[a_1\ a_2\ ...\ a_9]'$。

9. 線性迴歸(取代學習)：線性迴歸自變數 X 是 200 幅訓練影像的卷積特徵值，X 矩陣的大小是 200x9。依變數 Y 則是這 200 幅影像的分類標籤，Y 矩陣的大小是 200x1，前 100 個數值為 0，後 100 個數值為 1。使用 $A=(X^T*X)^{-1}*(X^T*Y)$算出 9x1 的係數矩陣。若使用 Matlab，A = inv(X'*X)*(X'*Y) 。亦可用 A= X\Y 替代。

10. 測試(混淆矩陣)：

    直接套用訓練集的 X 與 Y 矩陣資料做測試。首先用 Yp=X*A 獲得這 200 幅影像的迴歸預測值 Yp。如果該影像的 Yp 值低於 0.5，分類結果為 0，若 Yp 值高於 0.5，分類結果為 1。以'0'與'1'數值的分類為例，請把前 100 個 Yp 數值中，低於 0.5 的數量填入混淆矩陣(1,1)的位置，這代表'0'被正確判斷的次數。請把前 100 個 Yp 數值中，高於 0.5 的數量填入混淆矩陣(2,1)的位置，這代表'0'被誤認為'1'的次數。 請把後 100 個 Yp 數值中，高於 0.5 的數量填入混淆矩陣(2,2)的位置，這代表'1'被正確判斷的次數。請把後 100 個 Yp 數值中，低於 0.5 的數量填入混淆矩陣(1,2)的位置，這代表'1'被誤認為'0'的次數。

    表 1 是'0'與'1'數值分類的混淆矩陣實例，這個例子的分類「正確率(accuracy)」是 98%=(97+99)/(100+100)。

表 1: 混淆矩陣，以'0'與'1'數值影像辨識為例

混淆矩陣		實際的類別	
		'0'	'1'
預測的類別	'0'	97	1
	'1'	3	99

11. 隨機測試：從這 200 張影像任意選取(用 rand()函式)15 張，評估其效果。每個圖的標題含預測的數值，以及隨機選取的影像編號(括弧內)。若答對，文字是藍色的；若答錯，文字是紅色的。如**圖 2** 的例子：

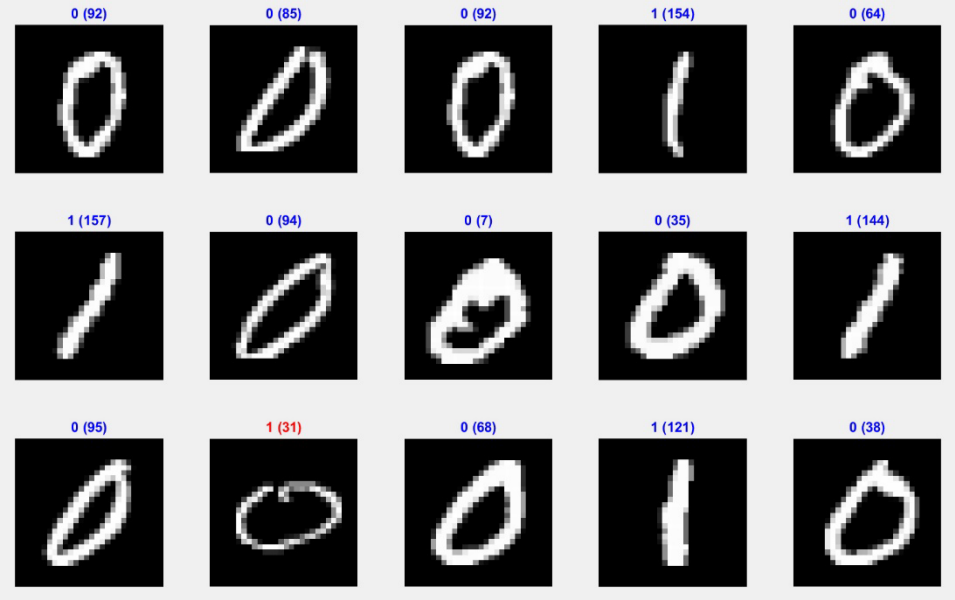

圖 2: 從 1~200 號影像隨機選 15 張顯示辨識成果，以'0','1'為例。

12. **加分題：**用梯度下降法，訓練分類器(不訓練卷積濾鏡)

(1) 將步驟 9 的 X 作為訓練輸入資料，Y 是分類目標值。

(2) 設定學習率 lr=0.001。權重向量 W 取代步驟 6 中的 A 矩陣，將初始的 W 權重向量的 9 個數值皆設為 0。

(3) 建立一個跑 1000 次的迴圈，每跑一次迴圈，都會更新一次均方誤差(Mean Square Error, MSE)、誤差梯度向量 G、權重增量向量 dW、以及權重向量 W。訓練批次大小為訓量樣本數，因此迴圈次數即為 epoch 次數。

(4) 均方誤差(也就是損失函式)：

$$E = \frac{1}{m}\sum_{i=1}^{m}(Y_i - Yp_i)^2 = \frac{1}{m}\sum_{i=1}^{m}(Y_i - X_i * W)^2 = mean((Y - X * W)^2)$$

m: 輸入的資料數量

Y_i: 分類目標資料的第 i 筆

Yp_i: 分類預測結果的第 i 筆

X_i: 訓練影像特徵值的第 i 筆

W: 權重向量

(5) 誤差梯度向量 G:

$$G = \frac{1}{m}\nabla E(W)$$

$$= -\frac{1}{m}\sum_{i=1}^{m}(Y_i - Yp_i)X_i = -\frac{1}{m}\sum_{i=1}^{m}(Y_i - X_i * W)X_i \ = -mean\,(\ldots..)$$

(6) 權重增量向量:

$$dW = \Delta W = -lr\nabla E(W) = -lr * G$$

(7) 權重向量更新:

$$W = W \ + dW$$

(8) 繪製 Epoch 與均方根誤差(MSE)的關係圖(如**圖 3**)

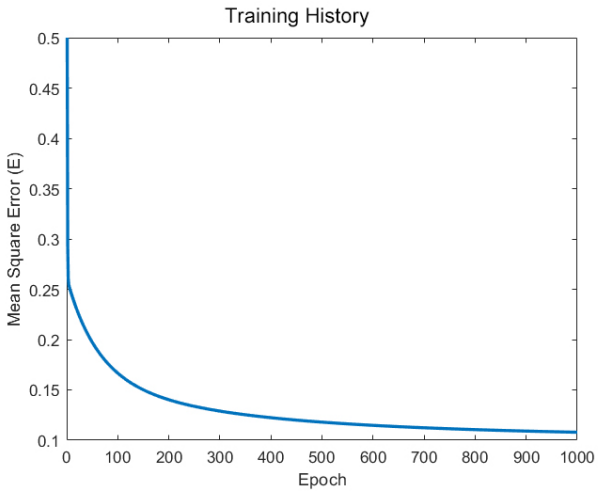

圖 3: Epoch 與均方根誤差(MSE)的關係圖。

(9) 預測結果:

與步驟 10 相同，用 X 與 Y 測試優化後的權重向量 W，所獲得的分類結果 Yp。混淆矩陣為何？辨識的正確率有多高？(註：未必比步驟 10 的線性迴歸高)

(10) 隨機測試:

與步驟 11 相同，隨機挑 200 幅訓練影像中的 15 幅作測試。例如**圖 4**。

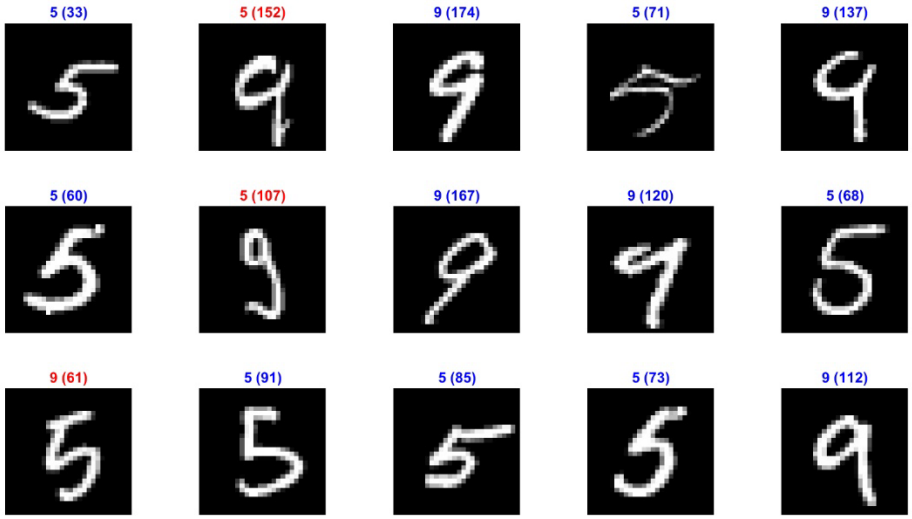

圖 4: 隨機選 15 張訓練影像顯示辨識成果，以'5','9'為例。

備註:

(1) Matlab 可能用到的函式: clc, clear, close, input, imread, im2double, imresize, imfilter, blkproc, mean, repmat, max, find, length, round, disp, figure, plot, num2str, subplot, imshow, title, xlabel, ylabel

(2) Python 可能用到的函式: random.uniform, random.randint, shape, cv2.imread, cv2.resize, np.sum, np.reshape, np.where, np.concatenate, np.tile, np.array, np.arrange, np.stack, np.append, np.pad, plt.figure, plt.subplot, plt.imshow, plt.plot, plt.xlabel, plt.ylabel, plt.show

(3) 注意運算時兩個陣列的大小以及形狀

(4) 做線性迴歸時是矩陣乘法不是點對點相乘

程式語言：可用 Matlab, Python, C++, Java, VB。

繳交內容：附上程式碼，程式需詳細註解，不用附訓練影像集。程式貼入 Word 檔，連同執行結果製作一個以 HW3_學號命名的 PDF 檔。將所有檔案放在以 HW3_學號命名的檔案匣，打包成 zip 檔上傳。

注意：不得使用深度學習或類神經網路套件。

繳交期限：111 年 12 月 30 日 24:00 前上傳至 Moodle 2 作業區。

有問題可請教色彩所劉光智同學(M11125007@mail.ntust.edu.tw)